Institute for Scientific Computing Research

Laboratory Directed Research and Development Project Research Summaries

Enabling Large-scale Data Access

Terence Critchlow
Center for Applied Scientific Computing

Summary

The web is the preferred method for distributing scientific data because it is easy to develop a customized web interface to data that provides scientists from around the world access to colleagues' results with the click of a mouse. This revolution has the potential to advance scientists' ability to perform research by enabling large-scale data exploration, improving communication among groups, increasing academic scrutiny of results, reducing duplication of effort, and encouraging collaborations. Unfortunately, these goals are floundering because scientists are overwhelmed by the hundreds of custom interfaces they must use to access the data.

This project is developing an architecture capable of identifying, categorizing, and wrapping (i.e., writing a class to interact with) these interfaces, allowing us to provide scientists a single interface to access hundreds of data sources. This interface will simplify scientists' interaction with the data and enable them to answer more complex questions than currently possible. Our proposed architecture uses service class descriptions to describe interfaces that are of interest to our scientists, a spider (i.e., a program that parses web pages and follows html links) capable of identifying instances of these interfaces when they are encountered, an interface description that is generated by the spider detailing how to interact with an interface of interest, and a wrapper generator to take the interface description and generate code capable of interacting with the interface. For the wrapper generator, we are expanding on the Xwrap Elite program developed by our collaborators at Georgia Institute of Technology.

This project will implement the architecture through a series of increasingly complex prototypes. Each successive prototype will improve access to real-world data sources, with the final one capable of categorizing several dozen complex, scientific data sources. The resulting wrappers will be accessed by our programmatic collaborators through a customized graphical query interface. This approach will allow us to evaluate alternative strategies in a realistic environment while quickly transferring the technology to our programmatic collaborators.

In FY02, we (1) wrote an initial specification of the interface description format, which will be used to describe an interface once spider has determined how to interact with it; (2) designed an extension to the existing XWrap Elite architecture to generate wrappers for complex interfaces; and (3) wrote a simple spider that takes a set of input pages as input, and goes through each page sequentially, identifying all of the links contained on the page and queuing them for traversal. While the current implementation of the spider is not yet capable of categorizing web pages, it is able to identify those pages that utilize web forms to provide data input.

For FY03, we will (1) write a white-paper specification of the service-class descriptions; (2) complete initial development of the XWrapComposer program; (3) create a version of the web spider program capable of automatically categorizing simple interfaces; and (4) provide our collaborators with access to new data sources through prototype inter

Terence Critchlow
Center for Applied Scientific Computing

Summary (continued)

face. This work supports all Laboratory missions by providing scientists direct and efficient access to more external data such as scientific publications, chemistry databases, material descriptions, production techniques, weather data, and urban planning information.

Overcoming the Memory Wall in SMP-Based Systems

Bronis R. de Supinski, Andy Yoo, Sally A. McKee, Frank Mueller, and Tushar Mohan

Center for Applied Scientific Computing

Summary

Both CPU and memory speeds are increasing at exponential rates, as expressed in Moore's Law. Unfortunately, memory hardware is slower than CPUs, and memory speeds are not increasing as rapidly as CPU speeds. For example, on snow, the ASCI White testbed system, an average of 87 floating point operations can be completed in the time required to load one operand from main memory. Even worse, CPU speeds are increasing faster than memory speeds; thus, the number of CPU cycles required to access memory is increasing. This divergence will exacerbate an existing problem for codes with large memory footprints, including the codes typically in use at LLNL: memory accesses dominate performance. Not only is the performance of many LLNL codes dominated by the cost of main memory accesses, but many current trends in computer architecture will lead to substantial degradation of the percentage of peak performance obtained by these codes. Many researchers anticipate a "Memory Wall" in which memory accesses imply an absolute performance limit, and improvements in CPU speed provide no performance benefit.

We are extending dynamic access optimizations (DAO), a promising set of techniques for overcoming the Memory Wall, to symmetric multiprocessors (SMPs); SMP-based systems are common at LLNL. DAO techniques have shown significant promise to overcome the Memory Wall without requiring complex source code changes. These techniques change the order or apparent locations of memory accesses from those generated by the issuing program to ones that use the memory system more effectively without changing the results. For example, altering the execution order can exploit memory hardware characteristics such as interleaved memory banks and hot dynamic random access memory (DRAM) rows, while techniques that alter the apparent location can significantly increase cache hit ratios. DAO mechanisms can reduce run times of memory intensive portions of programs by factors of two to an order of magnitude. Previous projects investigating DAO focus on uniprocessor systems and require special-purpose hardware. Although promising, DAO techniques for uniprocessors do not target the systems in use at LLNL. All major LLNL computing resources are clusters with SMP nodes. Thus, we need DAO techniques that support simultaneous access to the memory system by multiple processors.

Implementing DAO techniques for SMPs is more difficult than for uniprocessor systems, since multiple processors may access the same physical address through different apparent locations. This aliasing of actual physical address with other apparent physical addresses creates a remapping coherence problem, a variant of the cache coherence problem. SMP-aware DAO techniques must ensure that the results are consistent to those that occur without remapping. SMPs use hardware mechanisms, such as cache invalidations, to solve the cache coherence problem. However, these mechanisms are keyed on (apparent) physical addresses, and thus are not automatically invoked for all of the aliases created through remapping.

Bronis R. de Supinski, Andy Yoo, Sally A. McKee, Frank Mueller, and Tushar Mohan
Center for Applied Scientific Computing

Summary (continued)

We have designed three distinct mechanisms for solving the remapping coherence problem. Our most promising technique modifies the coherence controller to account for remapping. This solution requires that the coherence controller can access the alias translation mechanisms that support uniprocessor DAO techniques. Our hardware-based solution guarantees that coherence operations are invoked on all of the aliases of a (apparent) physical address, ensuring that the memory semantics are identical to those of the original machine. Our mechanism features a fast algorithm for identifying addresses that are not aliased. The algorithm executes concurrently with the operations already required to maintain coherence and access main memory. Thus, the cost of accessing unaliased locations remains essentially unchanged.

We have designed virtual pinned memory, a novel mechanism for providing true zero-copy message passing. Ordinarily, exchanging data between nodes in cluster-based systems requires memory copies to gather it from and scatter it into the user memory locations. Also, even contiguous user data must be copied into and out of system buffers in main memory or the user memory must be pinned to physical memory so that it can be copied to the network interface. Virtual pinned memory, derived directly from SMP-aware DAO mechanisms, eliminates these requirements. With virtual pinned memory, DAO-based scatter/gather and alias translation mechanisms copy user data directly to the network interface.

Finally, we continued our work on understanding application memory access regularity. Our novel tool that measures an application's regularity gathers statistics that strongly indicate what memory optimizations will improve the application's performance. This research, the subject of a Master's Thesis at the University of Utah, demonstrates that applications with irregular access patterns require our DAO mechanisms.

Visualization Streams for Ultimate Scalability (ViSUS): Monitoring Terascale Simulations in Real Time

Valerio Pascucci

Summary

Modern scientific simulations and experimental settings produce ever-increasingly large amounts of data that traditional tools are not able to visualize in real time, especially on regular desktop computers. Scientists are unable to interactively explore the data sets that they produce, which creates a frustrating slowdown in the overall process of scientific discovery. Use of innovative, high-performance visualization techniques that allow interactive display of very large data sets on simple desktop workstations and the monitoring (or steering) of large parallel simulations will have specific applications to several of the DOE's and LLNL's missions, including stockpile stewardship, energy and environment, nonproliferation, biology, and basic science, that use large-scale modeling and simulations.

The ViSUS system will implement unified scalable solutions allowing large data visualization on a single desktop computer, on a cluster of personal computers, and on heterogeneous computing resources distributed over a wide-area network. When processing terabytes of scientific data, our goal is to demonstrate an effective increase in visualization performance of several orders of magnitude in two major settings: (1) interactive visualization on desktop workstations of large data sets that cannot be stored locally; and (2) real-time monitoring of a large scientific simulation with negligible impact on the computing resources available. The main focus of the ViSUS research is in the development of a novel data-streaming infrastructure based on a suite of progressive and out-of-core visualization algorithms enabling the interactive exploration of scientific datasets of unprecedented size. The methodology aims to globally optimize the data flow in a pipeline of processing modules, with each module reading a multi-resolution representation of the input while at the same time producing a multi-resolution representation of the output. Multi-resolution methods provide the necessary flexibility to trade speed for accuracy in the visualization process. Maximum coherency and minimum delay in the data flow are achieved by extensive use of progressive algorithms (see figure below) that continuously map local geometric updates of the input stream into immediate updates of the output stream. We are developing a prototype implementation of this streaming infrastructure to demonstrate the practical flexibility and scalability of our approach.

During FY2002, we (1) implemented a new progressive slicing technique that computes planar cross sections of rectilinear grids with arbitrary orientation; (2) designed and implemented a new data layout allowing real-time geometric queries for rectilinear grids with up to 0.5 trillion nodes; (3) developed a novel multi-resolution representation that simplifies the real-time extraction of minimal adaptive models; (4) introduced the first subdivision method for unstructured meshes of any dimension and cells of virtually any type; and (5) tested the basic network components of our streaming infrastructure.

Valerio Pascucci

Summary (continued)

In FY2003 we will (1) build a prototype of a remote viewer that interactively slices data sets retrieved from a remote storage system; (2) implement a direct-streaming infrastructure prototype that connects the nodes of a scientific simulation to a set of data servers used for permanent storage; (3) parallelize the preprocessing of the input grids to achieve one order of magnitude improvement in the data reordering into our storage layout; (4) compute in external memory multi-resolution representations of triangulated surface meshes; (5) define and implement multi-resolution wavelet models to be combined with a new volumetric subdivision scheme; and (6) explore new image caching mechanisms for decoupling the display process from the image rasterization process.